

1 Recursive Weighted Least Square

The estimate of θ can be obtained based on measurements available at time k :

$$Z_k = H_k \theta + \epsilon_k \quad (1.1)$$

by using the least squares technique described earlier in class. Suppose additional measurements are obtained at time $k + 1$:

$$\zeta_{k+1} = h_{k+1} \theta + \varepsilon_{k+1}. \quad (1.2)$$

In recursive least squares, the problem we consider is how to compute the weighted least squares estimate of θ with all the measurements available at time $k + 1$, but without recomputing the weighted least squares solution directly, but rather by updating the estimate available at k .

We start by examining the weighted least squares estimate of θ with all the measurements available at time $k + 1$. Let

$$\begin{bmatrix} Z_k \\ \zeta_{k+1} \end{bmatrix} = \begin{bmatrix} H_k \\ h_{k+1} \end{bmatrix} \theta + \begin{bmatrix} \epsilon_k \\ \varepsilon_{k+1} \end{bmatrix} \quad (1.3)$$

$$Z_{k+1} = H_{k+1} \theta + \epsilon_{k+1}. \quad (1.4)$$

Use weighted least squares to find the new estimate:

$$\hat{\theta}_{k+1} = (H_{k+1}^T W_{k+1} H_{k+1})^{-1} H_{k+1}^T W_{k+1} Z_{k+1}, \quad (1.5)$$

where

$$W_{k+1} = \begin{bmatrix} W_k & 0 \\ 0 & w_{k+1} \end{bmatrix}. \quad (1.6)$$

Therefore

$$H_{k+1}^T W_{k+1} H_{k+1} = H_k^T W_k H_k + h_{k+1}^T w_{k+1} h_{k+1}. \quad (1.7)$$

Let $P_{k+1} = (H_{k+1}^T W_{k+1} H_{k+1})^{-1}$, then

$$P_{k+1}^{-1} = P_k^{-1} + h_{k+1}^T w_{k+1} h_{k+1}. \quad (1.8)$$

$$\hat{\theta}_{k+1} = P_{k+1} [H_k^T W_k Z_k + h_{k+1}^T w_{k+1} \zeta_{k+1}] \quad (1.9)$$

$$= P_{k+1} [P_k^{-1} P_k H_k^T W_k Z_k + h_{k+1}^T w_{k+1} \zeta_{k+1}]. \quad (1.10)$$

Since $\hat{\theta}_k = P_k H_k^T W_k Z_k$,

$$\hat{\theta}_{k+1} = P_{k+1} [P_k^{-1} \hat{\theta}_k + h_{k+1}^T w_{k+1} \zeta_{k+1}] \quad (1.11)$$

$$= P_{k+1} [(P_k^{-1} - h_{k+1}^T w_{k+1} h_{k+1}) \hat{\theta}_k + h_{k+1}^T w_{k+1} \zeta_{k+1}] \quad (1.12)$$

$$= \hat{\theta}_k - P_{k+1} h_{k+1}^T w_{k+1} h_{k+1} \hat{\theta}_k + P_{k+1} h_{k+1}^T w_{k+1} \zeta_{k+1} \quad (1.13)$$

$$= \hat{\theta}_k + P_{k+1} h_{k+1}^T w_{k+1} (\zeta_{k+1} - h_{k+1} \hat{\theta}_k) \quad (1.14)$$

The relation above provides a way to compute the weighted least squares estimate of θ using all the measurements available at time $k + 1$ by updating the estimate available at time k (that was obtained by using the measurements at time k). In other words, the new estimate is computed recursively by updating the previous estimate.

1.1 Computational Expense and Matrix Inversion Lemma

Equation (1.14) requires the inversion of a matrix

$$P_{k+1} = (H_{k+1}^T W_{k+1} H_{k+1})^{-1}, \quad (1.15)$$

which is a $n \times n$ matrix where $n = \dim(\theta)$. P_{k+1} needs to be computed every time new measurement is obtained. Since H_{k+1} and W_{k+1} gets bigger with k , the computation involved in the estimation gets more and more expensive.

We use a result known as the Matrix Inversion Lemma to reduce some of the computational burden. The Matrix Inversion Lemma states

$$\text{if } B^{-1} = A^{-1} + C^T D^{-1} C \quad (1.16)$$

$$\text{then } B = A - AC^T (CAC^T + D)^{-1} CA. \quad (1.17)$$

Applying the Matrix Inversion Lemma to Eq. (1.8), we get,

$$P_{k+1} = P_k - P_k h_{k+1}^T (h_{k+1} P_k h_{k+1}^T + w_{k+1}^{-1})^{-1} h_{k+1} P_k. \quad (1.18)$$

Note that P_{k+1} from Eq. (1.18) doesn't contain H_k and W_k . P_k is already calculated at time k , and h_{k+1} , w_{k+1} are in general much smaller than H and W . Therefore, it is less computationally expensive to obtain P_{k+1} using Eq. (1.18) than using Eq. (1.15), as all the involved calculations deal with the size as big as the new measurement.

1.1.1 Special Case - Scalar w_{k+1}

If the new measurement obtained at each time is a scalar, then P_{k+1} can be updated without any matrix inversion:

$$P_{k+1} = P_k - P_k h_{k+1}^T \frac{1}{\left(h_{k+1} P_k h_{k+1}^T + \frac{1}{w_{k+1}}\right)} h_{k+1} P_k \quad (1.19)$$

1.2 Example

Take the system ID example discussed in Lecture 1 (see *rwls.m*). The parameters are first estimated after collecting 5 samples of data ($k = 5$). Then estimates for a_d and b_d are recursively updated at every time as new measurements are obtained, till $k = 100$. Figure 1.1 show how the estimates change as a function of time index k , as they are recursively updated.

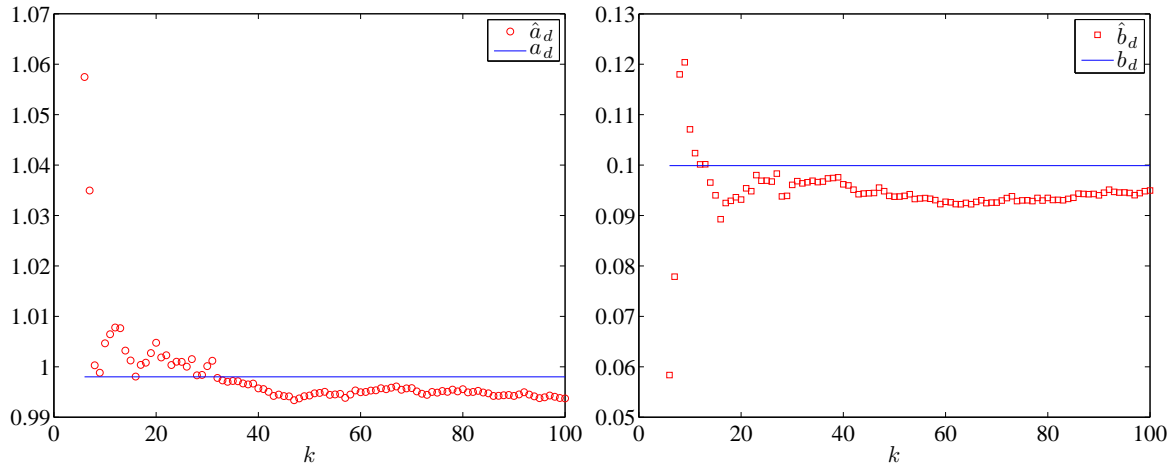


Figure 1.1: The estimate of \hat{a}_d and \hat{b}_d ($5 < k$)

rwls.m

```

% System ID example
% -Weighted least square
% -Recursive weighted least square
clear all
close all
clc

randn('state',sum(100+clock))
n=500;
% parameters of the continuous time model
a = -0.2;
b = 10;
T = 0.01; %sampling period, in second;
forcing_freq = 8; %in Hz
% parameters of the discrete time model
ad = exp(a*T);
bd = b*(ad-1)/a;
%initial condition
xinit = 0.5;
N = round(1/T); %number of data points collected in a single experiment
time_vec = [0:1:N-1]*T;
std_dev_u = 0.01;
std_dev_x = 0.02;
u_vec = sin(2*pi*forcing_freq*time_vec); %input - unit amplitude sinusoid

%
% Weighted least square with the first 5 measurements
%
W = diag(randn(5,1).^2);
x_prev = xinit;
for ii=1:5
    u_actual = u_vec(ii) + std_dev_u*randn(1,1);
    x_next = ad * x_prev + bd*u_actual + std_dev_x*randn(1,1);
    z(ii,1)=x_next;

```

```

        H(ii,:)=[x_prev u_vec(ii)];
        x_prev = x_next;
    end;
theta_hat = inv(H'*W*H) * H' * W * z;

theta_hat0 = theta_hat;
%
% Recursive weighted least square
%
theta_hat_rwls = zeros(2,N);

for ii=6:N
    w = 2;
    W_new = [W zeros(size(W,1),1);zeros(1,size(W,2)) w];
    u_actual = u_vec(ii) + std_dev_u*randn(1,1);
    x_next = ad * x_prev + bd*u_actual + std_dev_x*randn(1,1);
    z(ii,1)=x_next;

    h = [x_prev u_vec(ii)];
    H_new = [H; h];

    Pk = inv(H'*W*H);
    % Pk_new = inv(H_new'*W_new*H_new);
    % From matrix inversion lemma,
    Pk_new = Pk - Pk*h'*(h*Pk*h' + 1/w)^(-1) * h*Pk;

    theta_hat_new = theta_hat + Pk_new*h'*w*(x_next - h*theta_hat);
    theta_hat_rwls(:,ii) = theta_hat_new;

    % update
    x_prev = x_next;
    H = H_new;
    W = W_new;
    Pk = Pk_new;
    theta_hat = theta_hat_new;
end

%
% Plot the result
%
k = 6:100';

set(0,'DefaultAxesFontSize',20,'DefaultAxesFontName','Times')
set(0,'DefaultTextInterpreter','latex')

figure(1)
plot(k,theta_hat_rwls(1,6:N),'ro',k,ad*ones(size(k)))
xlabel('k');
legend('â_d','a_d')
set(legend(),'interpreter','latex')
print -depsc2 -r600 ad.eps

figure(2)
plot(k,theta_hat_rwls(2,6:N),'rs',k,bd*ones(size(k)))
xlabel('k');
legend('b̂_d','b_d')
set(legend(),'interpreter','latex')
print -depsc2 -r600 bd.eps

```